Learning from our keyring: What do our PGP keys say about the project?

Gunnar Wolf

Debian Project

DebConf 16
Capetown, South Africa, 2016-06-04

Contenidos

Once upon a time in Portland...

- This work dates back to our (keyring-maint's) presentation in DebConf14
- We were pushing to migrate away from short (<2048 bit) keys... but the progress was too slow
- Needed to show people how we were stagnating on something widely regarded as urgent
- ...Numbers speak for themselves
  - Graphs help us get the point accross

Just how deep was our problem?

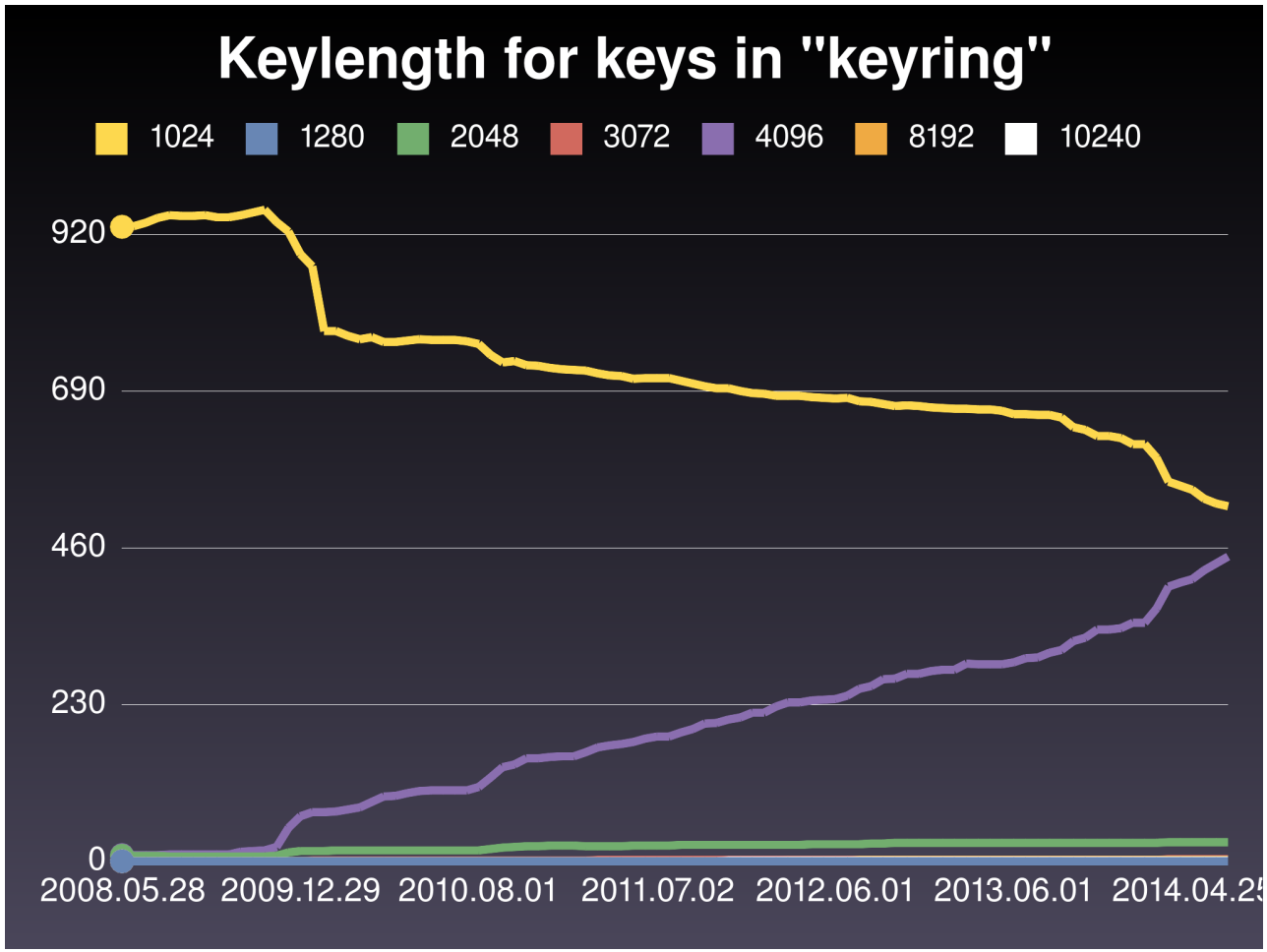


Figure: The situation as presented in DC14

And it worked...

- The rest of the project saw we sorely needed to deprecate short keys
- It was quite hellish for the team
  - 258 key replacements handled throughout less than half a year
- We perceived it as a successful transition... Although not exempt of problems
  - 287 keys (35 DMs, 252 DDs) not handled (thus removed)
  - 18 months later, 195 DD accounts still have a removed key

How does the void really look like...

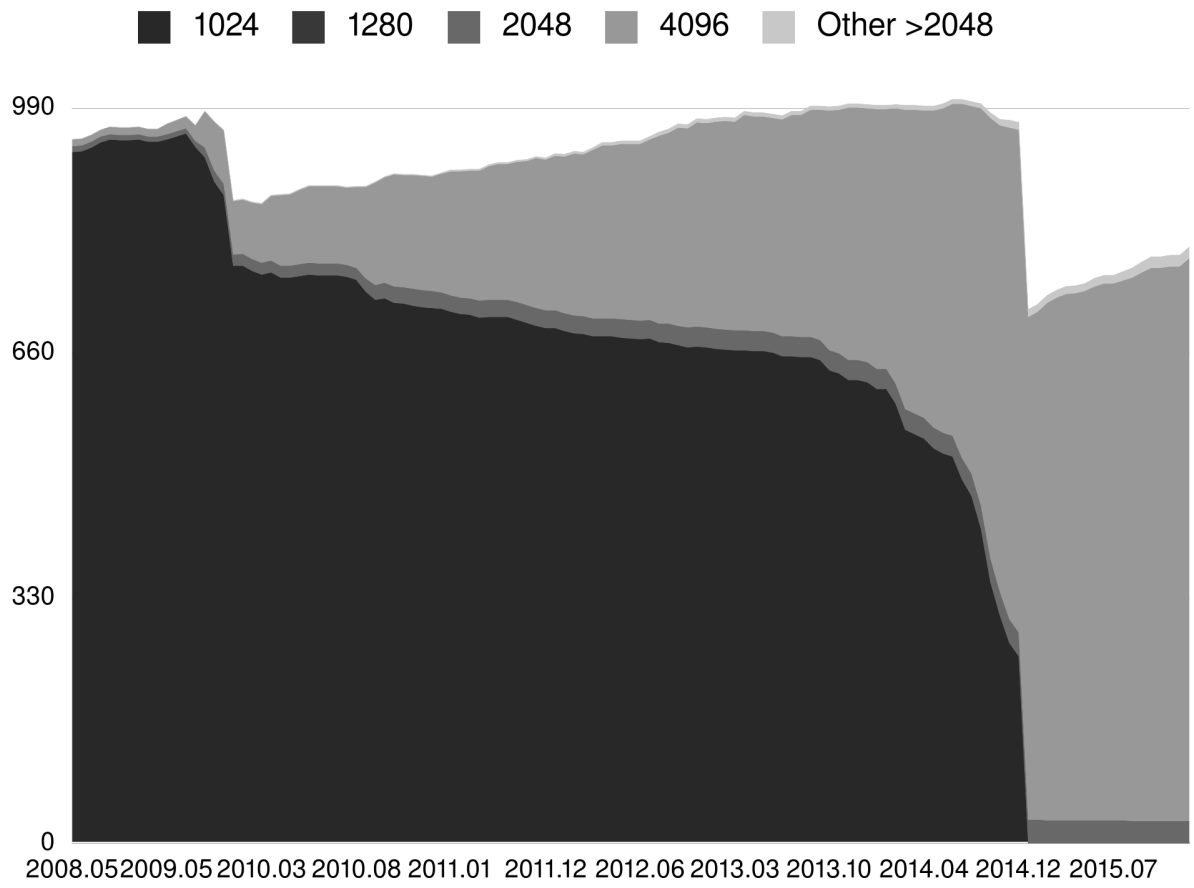


Figure: Drop in active DD keys after the <2048 bit removal

Numbers today

Debian Developer, uploading	816
Debian Developer, nonuploading	18
Debian Maintainer	236
Role keys	6

Contenidos

Given I am already analyzing stuff...

- Wrote a series of scripts to query+graph several aspects... Went on querying the data set
- What further measurements can be made to a keyring?
  - Finding the evolution of our *strong set* WRT the whole keyring
  - Surprise: Mostly stable over the years
  - Even more so discounting the *jitter* of 2014's changes

A stable proportion of strong set

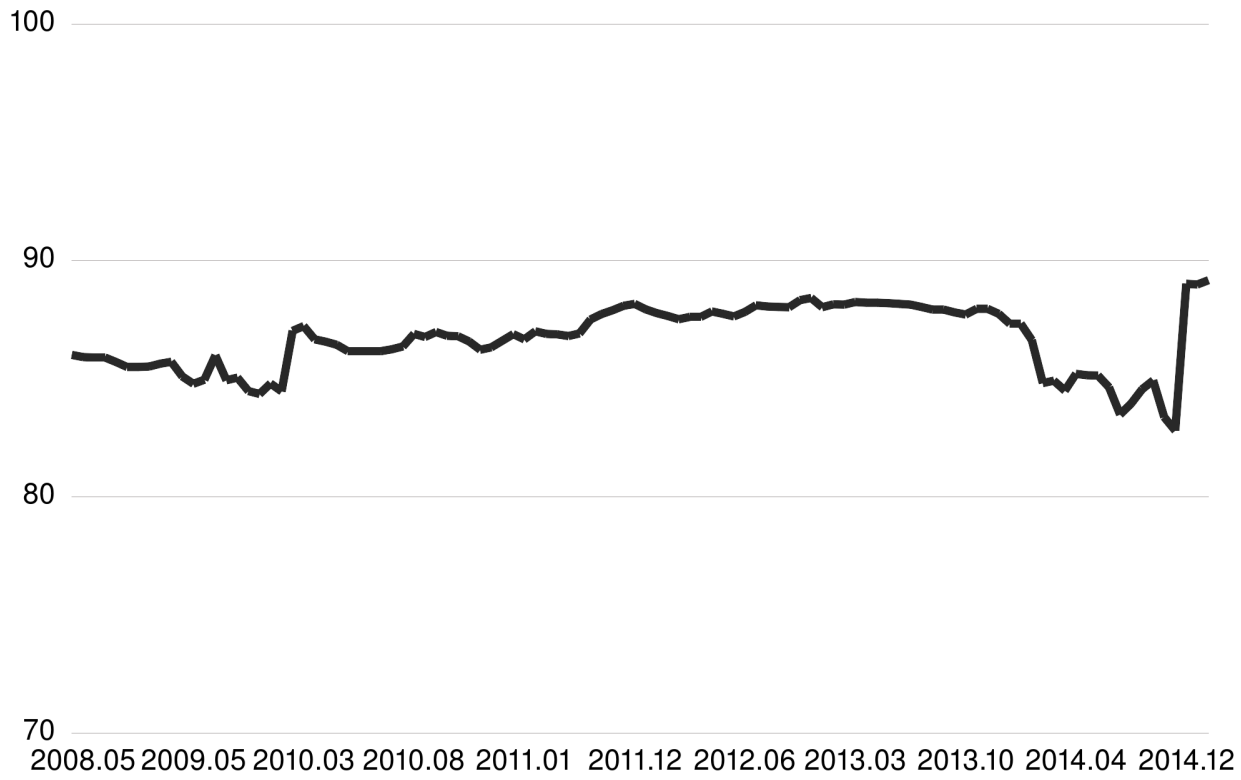


Figure: Strong set remains 82-89% of the keyring

Out of curiosity, the shape of the keyring

- Played with giving the keyring to `graphviz`
  - Might not be the best tool
  - Graph orientation and general shape is not *stable*
  - ...But the results are interesting nonetheless!
  - Excuse the ugliness when presenting... :-P
- Keys are nodes, signatures are edges
- Of course, it looks like a simple, useless blob...

The current, simple, boring blob: DDs, 2016.06.19

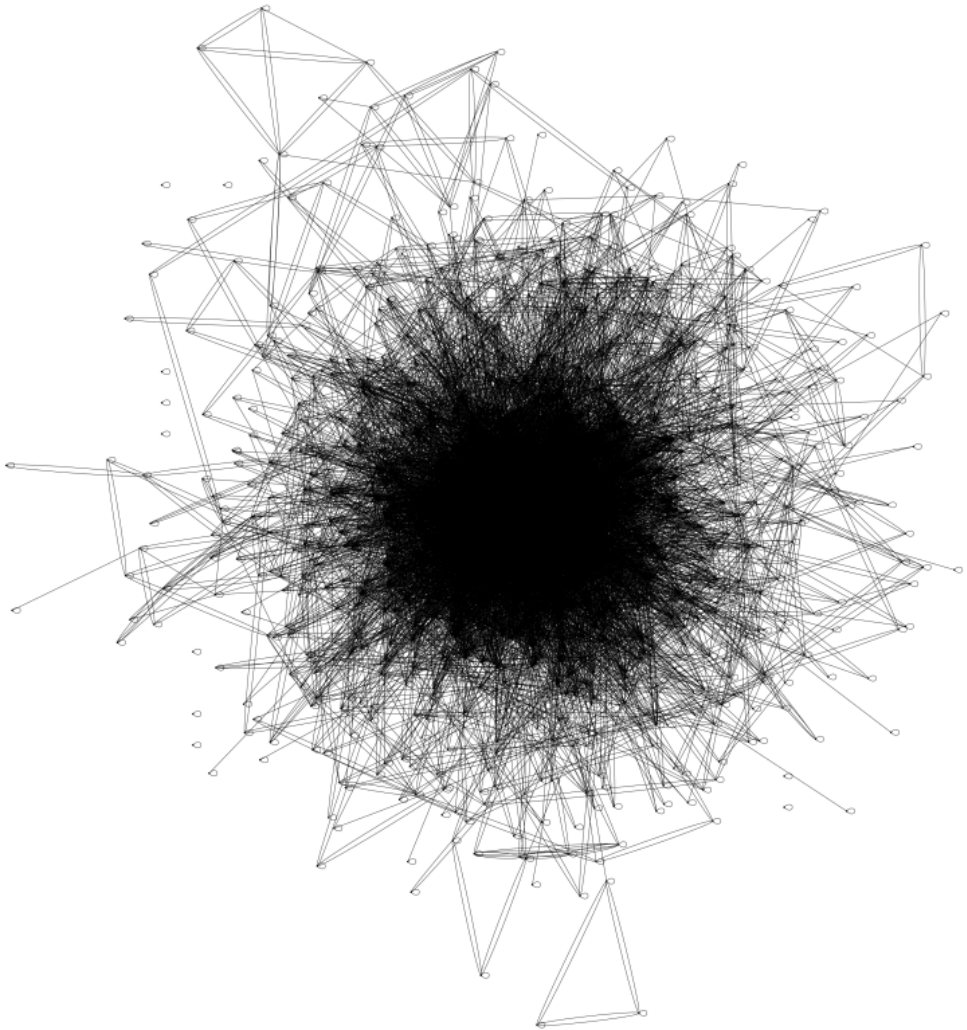
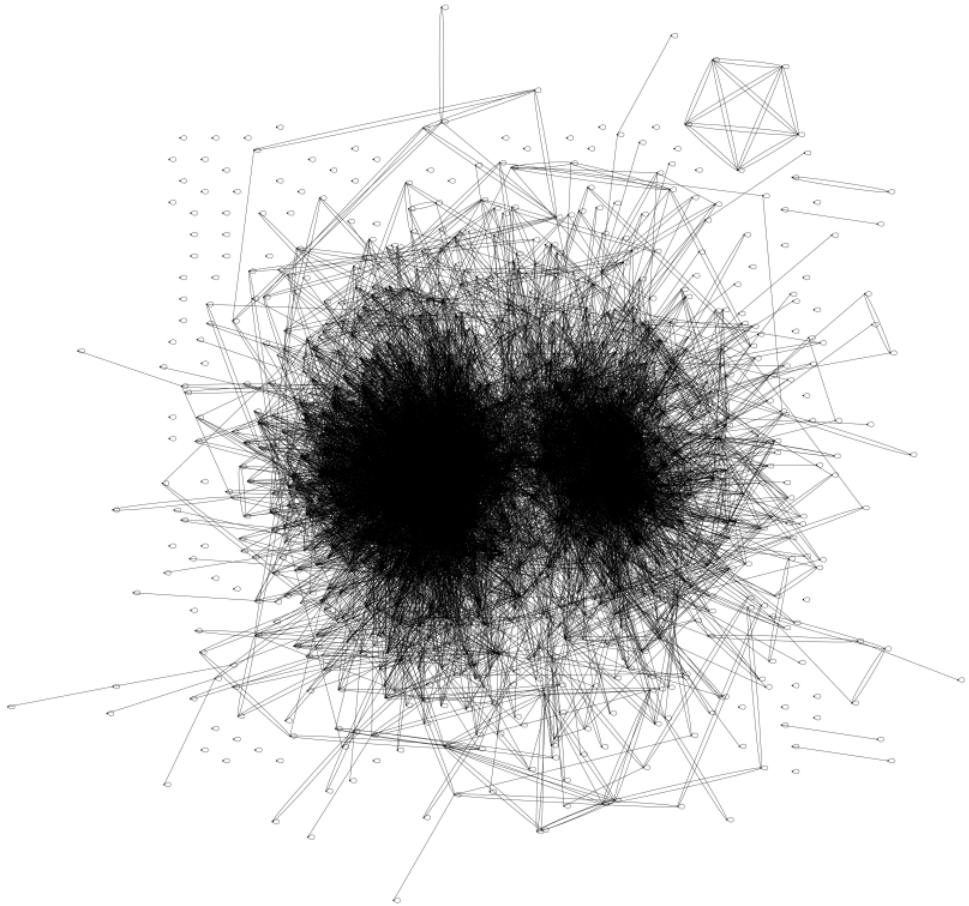


Figure: Our WoT — A maze of twisty passages, all alike

A *fun* blob: Debian Developers, mid 2014



keyring at 2014.07.28 (1002 keys)

Figure: It's ALIVE!!!

Given we are in Git, how *did* it look?

- What does this split mean?
- Why did it appear?
- Where does it come from?
- How did it get there?
  - What does that even mean?!

Evolution of the keyring

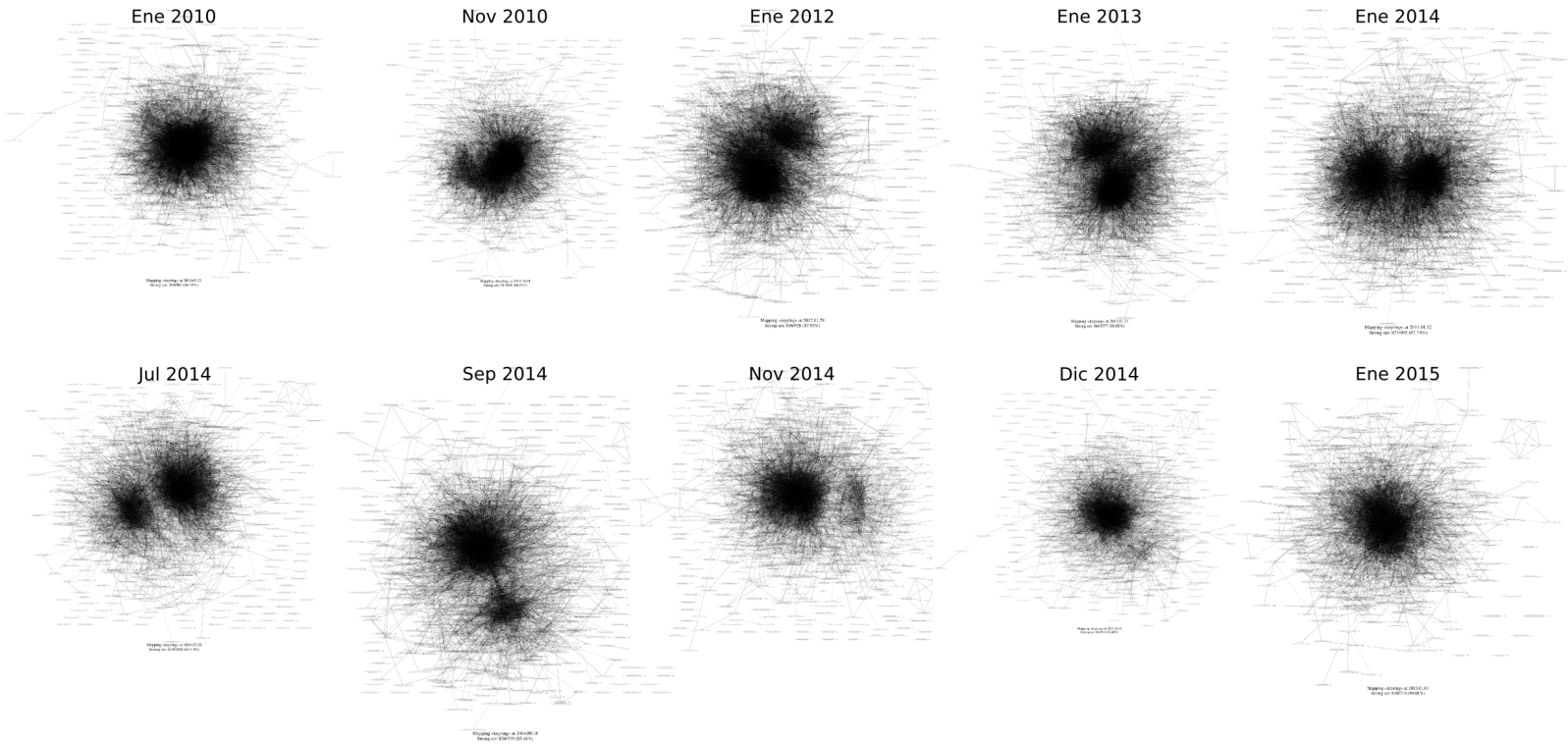


Figure: Top row: Yearly snapshots, 2010–2015; bottom row: ≈ bimestral snapshots; July 2014–January 2015

Contenidos

Starting point

Why stop there?

Keyring aging: A hypothesis

Signature expiration

Asymetric signatures

Further ideas

As for DC16...

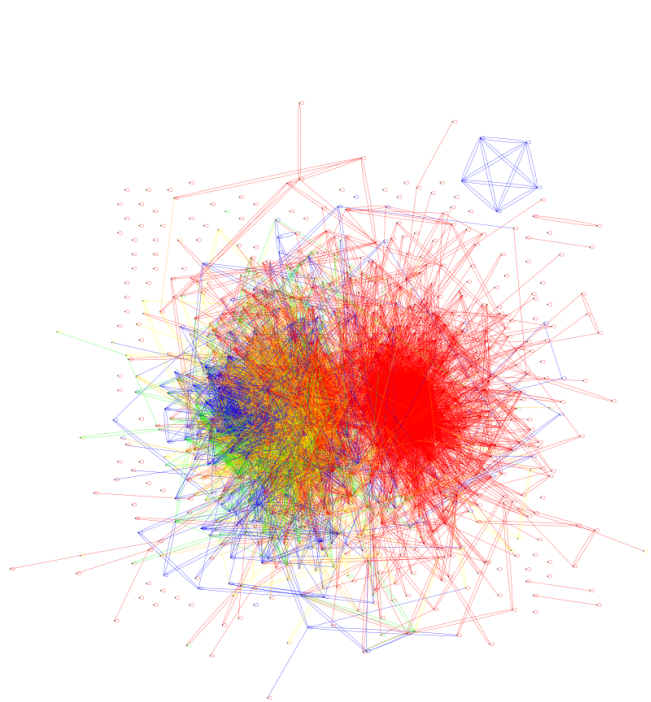
Hypothesis: Keyring aging?

- Leading to, and mostly during 2014, a huge portion of our keyring was replaced
  - One of the "blobs" marks older keys, the other new replacements?
  - But why the split began as early as 2011?
  - Note that nodes are grouped by their *cross-signatures* not by the key age (hence a 1024D key could be in the "younger" group and be expired!)
- Or it marks a *generation* of DDs, slowly going MIA?

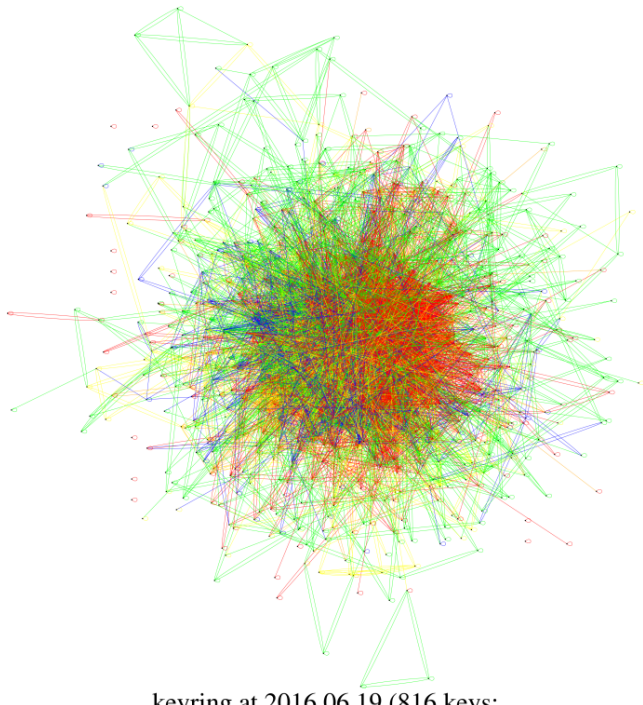
Graphs are nice, so...

- Colored graphs must be even better!
- Color key:
  - Nodes are irrelevant (`point`), only edges are important
  - Edges represent key signatures; color denotes signature age WRT the point in time the snapshot was *taken*
  - Blue: Less than one year
  - Green: 1 to 2 years
  - Yellow: 2 to 3 years
  - Orange: 3 to 4 years
  - Red: over 4 years old

Same two keyrings: 2014.07.28 and 2016.06.19



keyring at 2014.07.28 (1002 keys:
1660 <1yr 699 <2yr, 1537 <3yr, 2386 <4yr, 7313 older)



keyring at 2016.06.19 (816 keys:
1501 <1yr 2785 <2yr, 1962 <3yr, 970 <4yr, 5491 older)

(a) Big, red, disconnected blob (b) Now, a more even distribution

Same ten-keyring snapshot

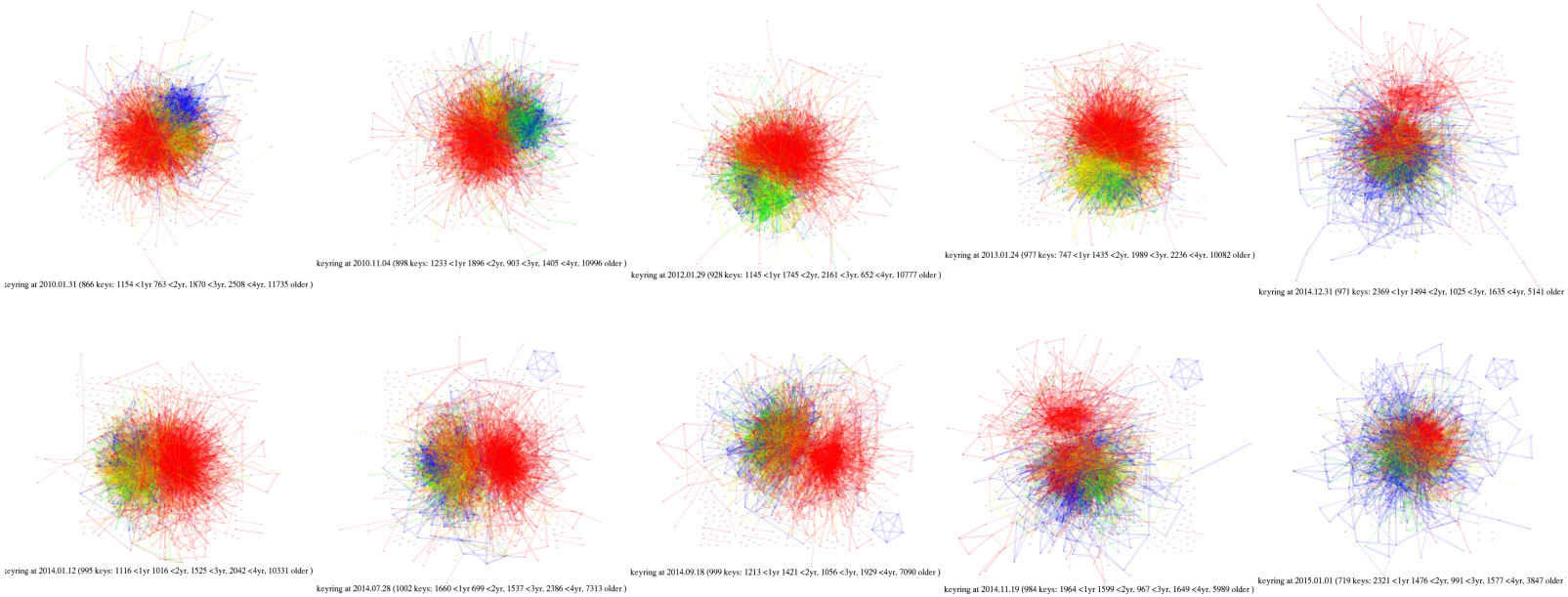


Figure: Over the years, the red blob manages to stay apart — until it fades and dies

- Seems to confirm the hypothesis

What comes to my mind

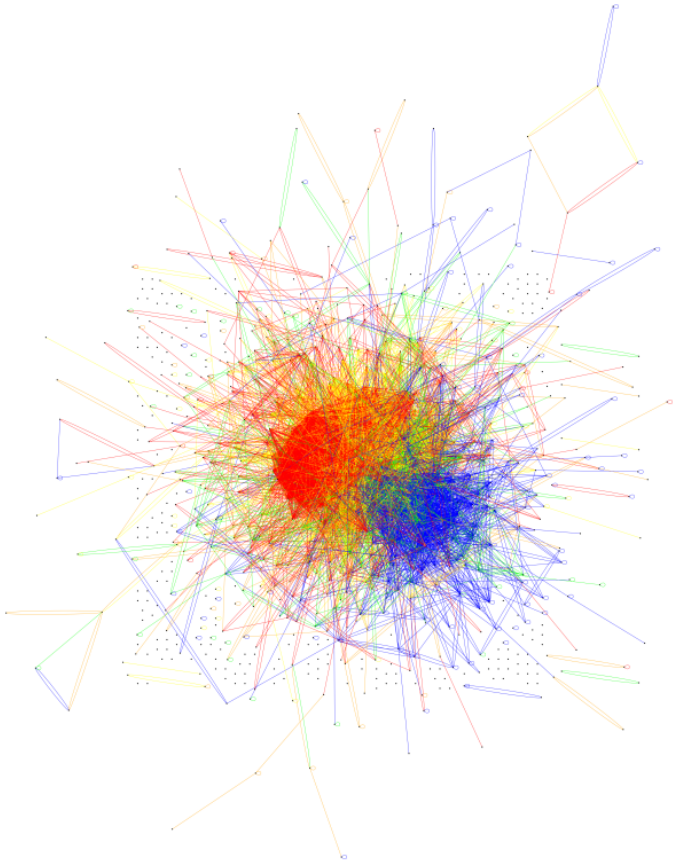
- Key signatures *do not imply friendship or trust*
  - Just trust that a given, identifiable party has control over a key pair
  - But it is an **important measure of trust** in our project
  - The only bit that links our electronic activities to our worldly identity
- ...How long should this trust last?
  - Do you still recognize everybody I have exchanged signatures with in the last decade?
  - Do you still vouch they control said key pair?
  - Do you have any grounds to believe nobody has vulnerated their security?

Contenidos

Looking at an auto-expiring keyring

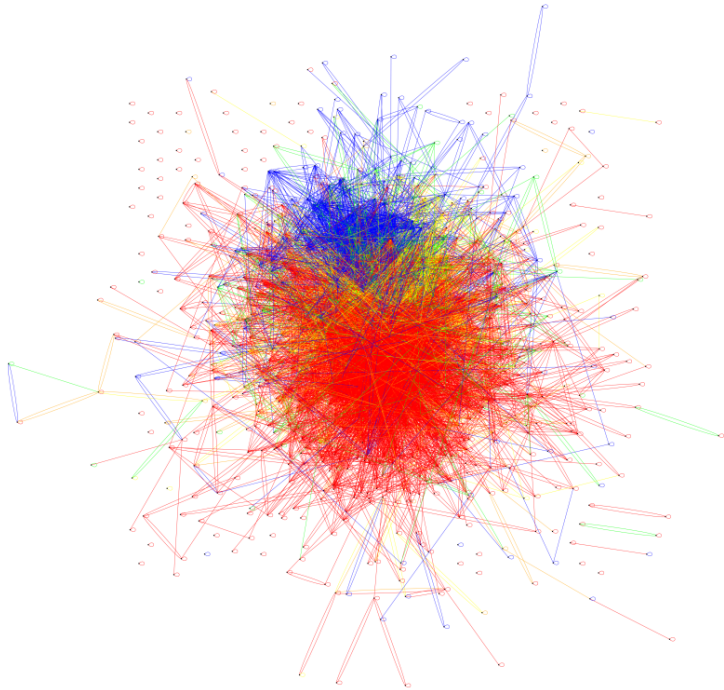
- If our *curatory* process were to regard all signatures over five years old as expired, how would the keyring look?
  - Note that is is just *for the sake of the excercise*
  - I'm not suggesting `keyring-maint` implements this
  - In fact, it's been outright discarded... But I have the graphs ;-) So lets play
- Main issues: How many people would *fall off* the strong (or reachable) set were we to discard old signatures
  - Would large *islands* be formed? Or just isolated dots

With/without expiring signatures, mid 2010



keyring at 2010.06.08, expiring 6691 sigs over 1825 days
(886 keys: 1633 <1yr, 866 <2yr, 1646 <3yr, 1121 <4yr, 5911 older)

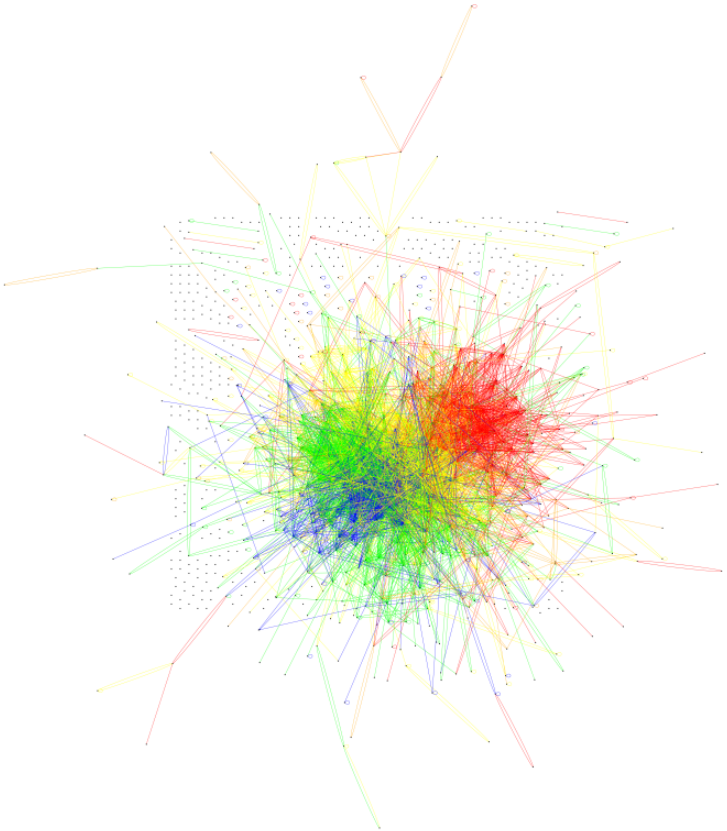
(a) Expiring >5yr old



keyring at 2010.06.08 (886 keys:
1633 <1yr 866 <2yr, 1646 <3yr, 1121 <4yr, 12602 older)

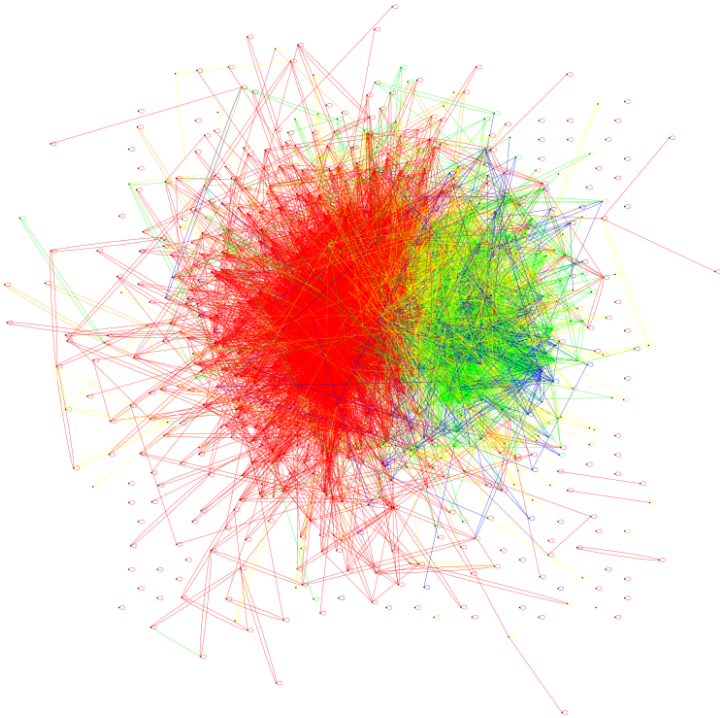
(b) Not expiring

With/without expiring signatures, mid 2012



keyring at 2012.06.01, expiring 9484 sigs over 1825 days
(946 keys: 1148 <1yr, 1802 <2yr, 2240 <3yr, 600 <4yr, 997 older)

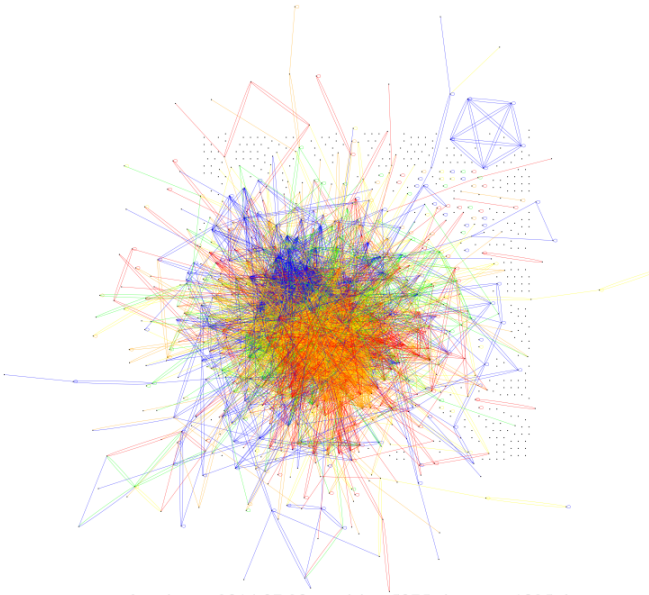
(c) Expiring >5yr old



keyring at 2012.06.01 (946 keys:
1148 <1yr 1802 <2yr, 2240 <3yr, 600 <4yr, 10481 older)

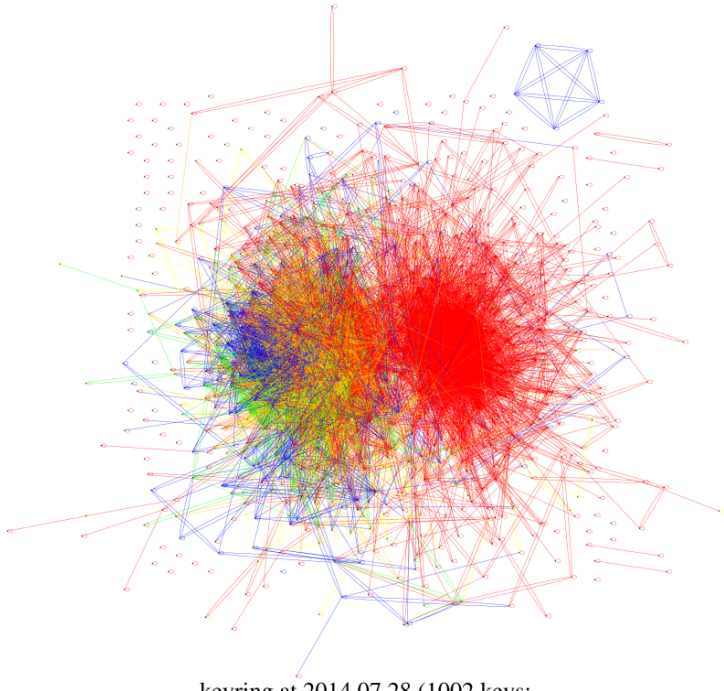
(d) Not expiring

With/without expiring signatures, mid 2014



keyring at 2014.07.28, expiring 5075 sigs over 1825 days
(1002 keys: 1660 <1yr, 699 <2yr, 1537 <3yr, 2386 <4yr, 2238 older)

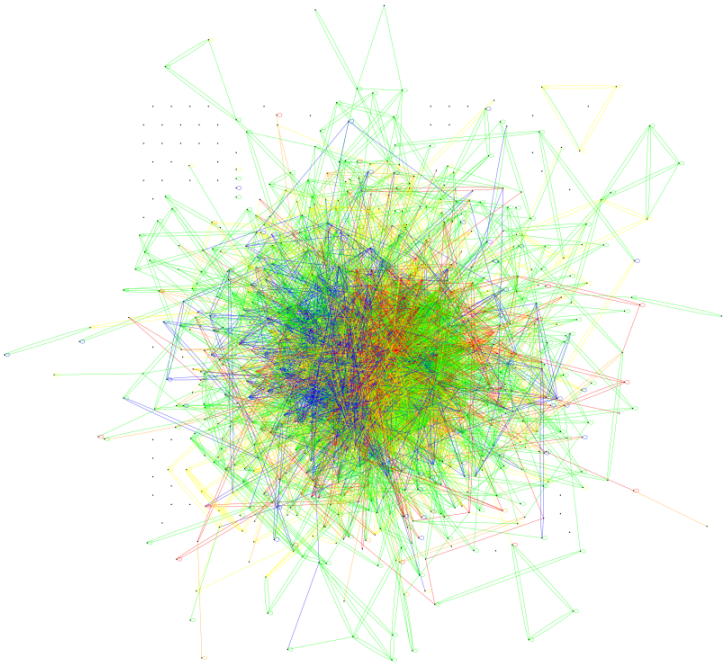
(e) Expiring >5yr old



keyring at 2014.07.28 (1002 keys:
1660 <1yr 699 <2yr, 1537 <3yr, 2386 <4yr, 7313 older)

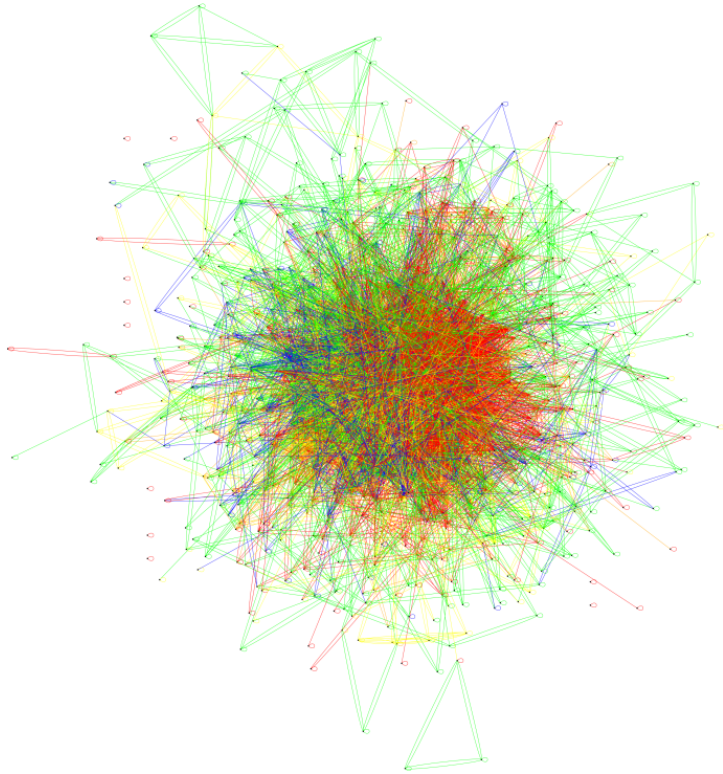
(f) Not expiring

With/without expiring signatures, today



keyring at 2016.06.19, expiring 3921 sigs over 1825 days
(816 keys: 1501 <1yr, 2785 <2yr, 1962 <3yr, 970 <4yr, 1570 older)

(g) Expiring >5yr old



keyring at 2016.06.19 (816 keys:
1501 <1yr 2785 <2yr, 1962 <3yr, 970 <4yr, 5491 older)

(h) Not expiring

Suggestion: Expire *your* signatures?


```
GPG(1)                                GNU Privacy Guard                                GPG(1)

NAME
       gpg - OpenPGP encryption and signing tool

SYNOPSIS
       gpg [--homedir dir] [--options file] [options] command [args]

DESCRIPTION
(...)
       --ask-cert-expire

       --no-ask-cert-expire
              When  making a key signature, prompt for an expiration time. If this option is not specified,
              the expiration time set via --default-cert-expire is used. --no-ask-cert-expire disables this
              option.

       --default-cert-expire
              The default expiration time to use for key signature expiration.  Valid values are "0" for no
              expiration, a number followed by the letter d (for days), w (for weeks), m (for months), or y
              (for years) (for example "2m" for two months, or "5y" for five years), or an absolute date in
              the form YYYY-MM-DD. Defaults to "0".
```

- Remember to do so also in `.caff/gnupghome/gpg.conf`

Contenidos

Starting point

Why stop there?

Keyring aging: A hypothesis

Signature expiration

Asymetric signatures

Further ideas

As for DC16...

What would asymetric signing be?

- Keysigning usually happens in pairs
  - I check your ID, you check mine, we agree to sign
  - ...Does not always happen that way, but *mostly*
  - What if we casually meet, but I didn't have a printed key on me? I can still sign yours...
- How often is this so?
  - What could it mean?

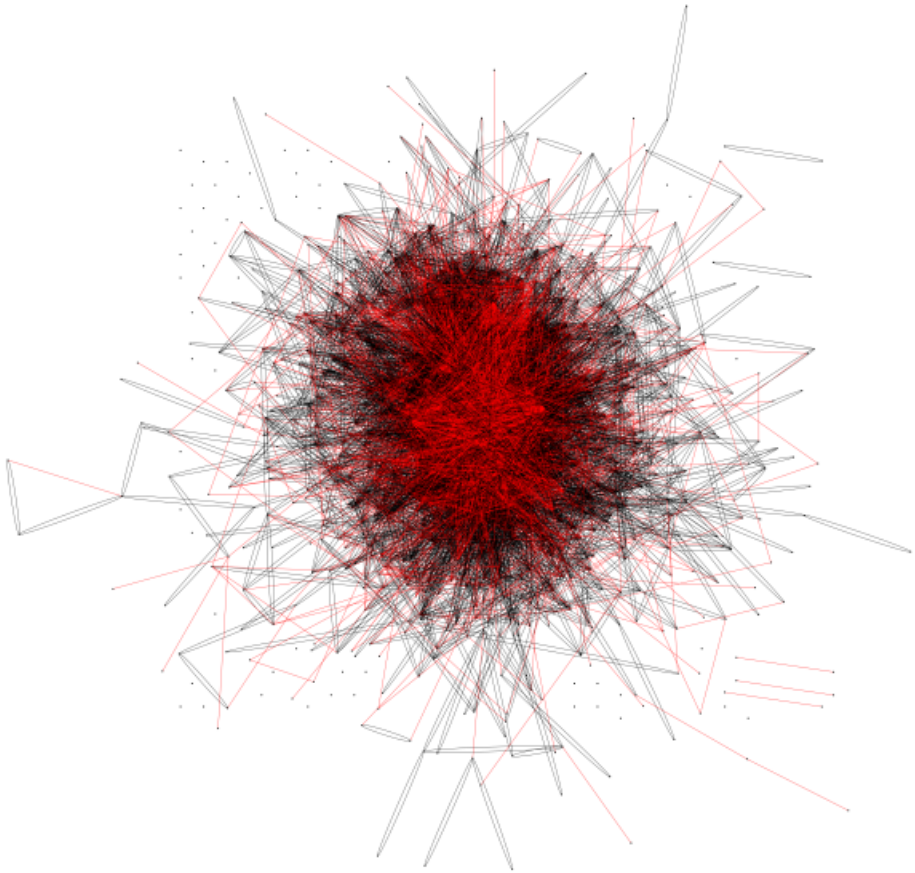
Wary of the big, bad KSP

- Earlier DebConfs: The biggest, baddests KSPs ever
- I personally got a couple of mails in the past:

```
1  Hi Gunnar,
2
3  While my key was in the DCn KSP, in the end I didn't
4  make it. But you still signed my key.
5
6  You know, that's bad practice and sloppy checking!
```

- Should this worry us?

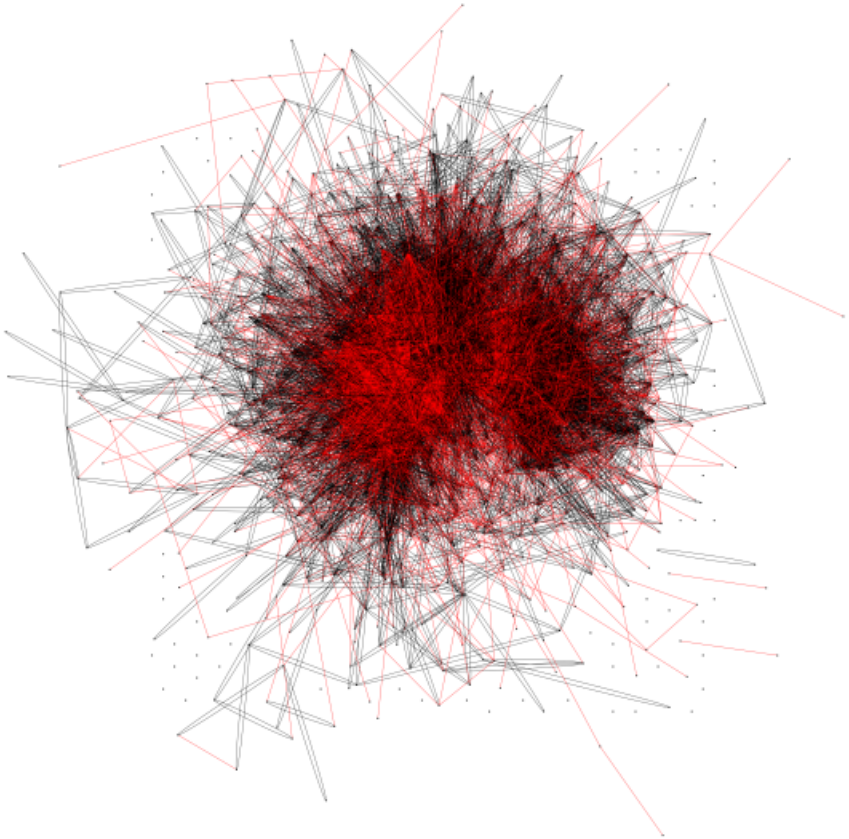
Peeks through time (1/4)



Asymetric signatures on keyring at 2010.06.08 (886 keys, 12600 mutual, 4382 single: 25.80% single)

Figure: Asymetric signatures in 2010

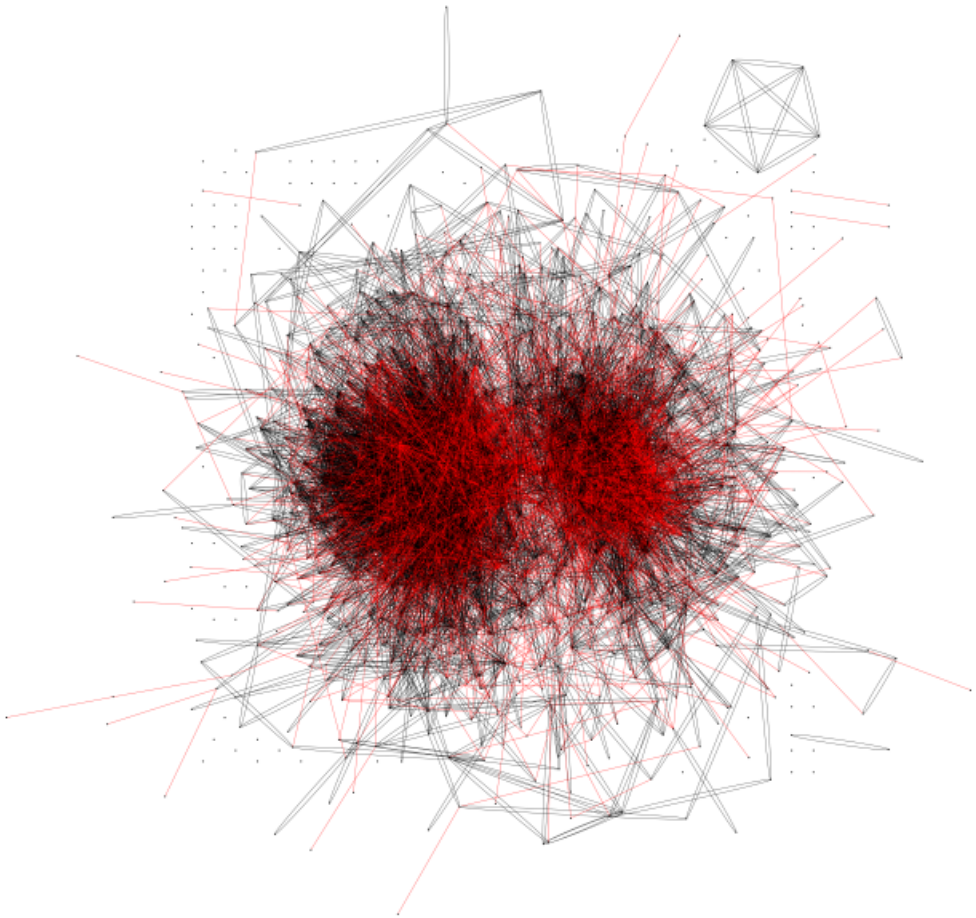
Peeks through time (2/4)



Asymetric signatures on keyring at 2012.06.01 (946 keys, 11064 mutual, 4261 single: 27.80% single)

Figure: Asymetric signatures in 2012

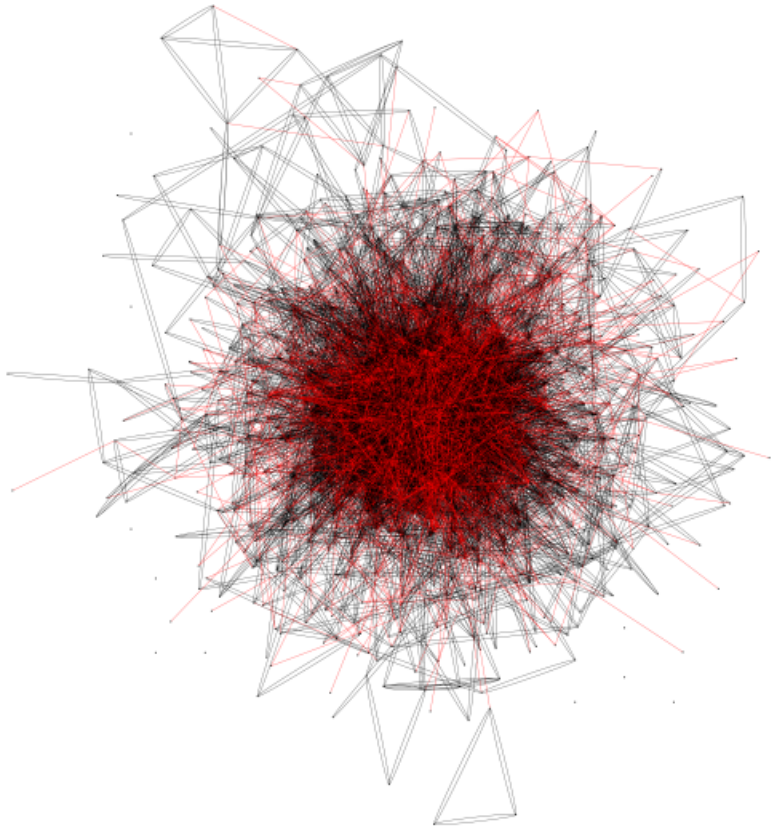
Peeks through time (3/4)



Asymetric signatures on keyring at 2014.07.28 (1002 keys, 9172 mutual, 3421 single: 27.17% single)

Figure: Asymetric signatures in 2014

Peeks through time (4/4)



Asymetric signatures on keyring at 2016.06.19 (816 keys, 8996 mutual, 2897 single: 24.36% single)

Figure: Asymetric signatures in 2016

Stable trends

- Max: 28.14%
- Average: 26.57%
- Min: 23.64%

Contenidos

Starting point

Why stop there?

Keyring aging: A hypothesis

Signature expiration

Asymetric signatures

Further ideas

As for DC16. . .

Other ideas to analyze

Exclusively for academic uses! (no policy changes in sight!)

- Algorithms used for signatures
- Number / shape of *islands*
- Analyze features the keys themselves, rather than whole keyrings
- Playing with the minimum *degree of connectedness*: Does one signature suffice? Two? Three? How much would the WoT *suffer* if we had stricter requisites?
- Identifying main hubs. How resilient is the WoT to withstand the loss of a hub?
  - Read: As simple as the replacement of a key
- Whatever ideas we can come up with :)

Contenidos

The KSP keyring

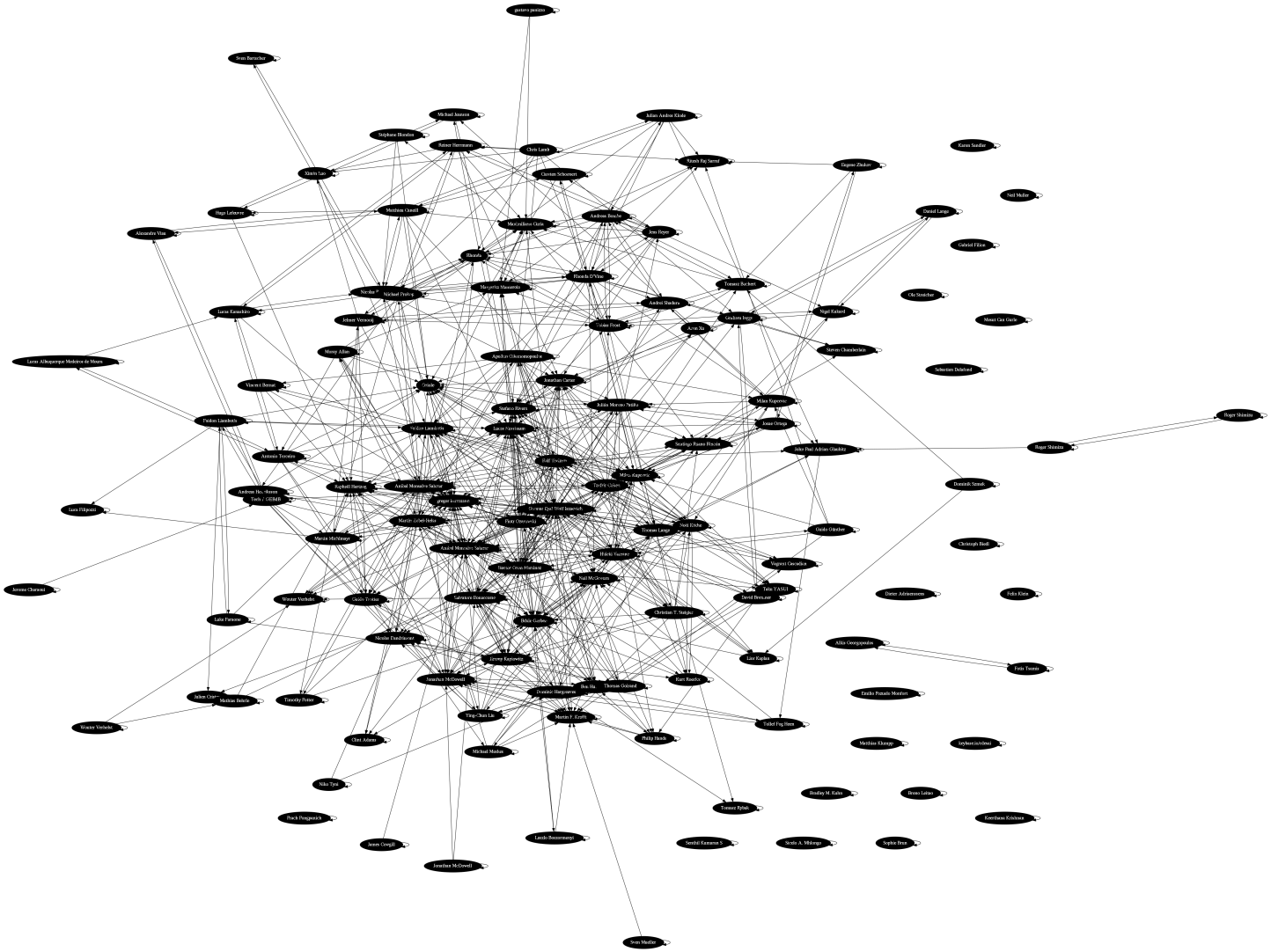


Figure: How is *our KSP* keyring structured?

Getting to your lucky number...


```
1
2  $ wget
       https://people.debian.org/~anibal/ksp-dc16/ksp-dc16.txt
3  --2016-06-30 23:26:51--
       https://people.debian.org/~anibal/ksp-dc16/ksp-dc16.txt
4  (...)
5  Saving to:  ksp -dc16.txt
6
7  ksp-dc16.txt
       100%[=======================================>]
       46.54K   138KB/s   in 0.3s
8
9  2016-06-30 23:26:52 (138 KB/s) -  ksp -dc16.txt  saved
       [47659/47659]
10
11 $ sha256sum ksp-dc16.txt
12 f5a470fa7abd521af677d1a212d3aca2180136b13323261d6e1cc316a1732bf1
        ksp-dc16.txt
```

Your lucky number...


```
f5a4 70fa 7abd 521a f677 d1a2 12d3 aca2
1801 36b1 3323 261d 6e1c c316 a173 2bf1
```